Hepatoprotective effects of systemic ER activation

BulkRNAseq - ER agonist treatment responses

Christian Sommerauer & Carlos Gallardo

25 July, 2023

```
# source and library import
source('code/00_helper_functions.R')
library(tidyverse)
library(Mfuzz)
library(ggvenn)
library(ggalluvial)
library(patchwork)
library(hypeR)
library(rrvgo)
library(scatterpie)
library(ggrepel)
```

```
# color palettes
colPals <- list()
colPals$conditions <- setNames(c('#44AA99', '#117733', '#88CCEE', '#332288', '#DDCC77', '#CC6677', '#AA4499', '#882255'),
                               c('CDf', 'HFDf', 'CDm', 'HFDm', 'DPN', 'DIP', 'E2', 'PPT'))
colPals$RdBu <- rev(RColorBrewer::brewer.pal(n=11, name = 'RdBu'))
colPals$UpDown <- setNames(colPals$RdBu[c(10,2)],
                           c('up', 'down'))
colPals$clusters <- setNames(c('#E6E6E6', '#B3B3B3', '#8C8C8C', '#4D4D4D'),
                             c('1', '2', '3', '4'))
```

NOTE

> For reproducibility of GO biological process PCA plot in semantic space (fig. 2C) run this code using `org.Mm.eg.db` v3.12.0 with R version 4.0.5. If this version is not used, enriched GO terms will still be the same but the distance between terms and clustering may vary.

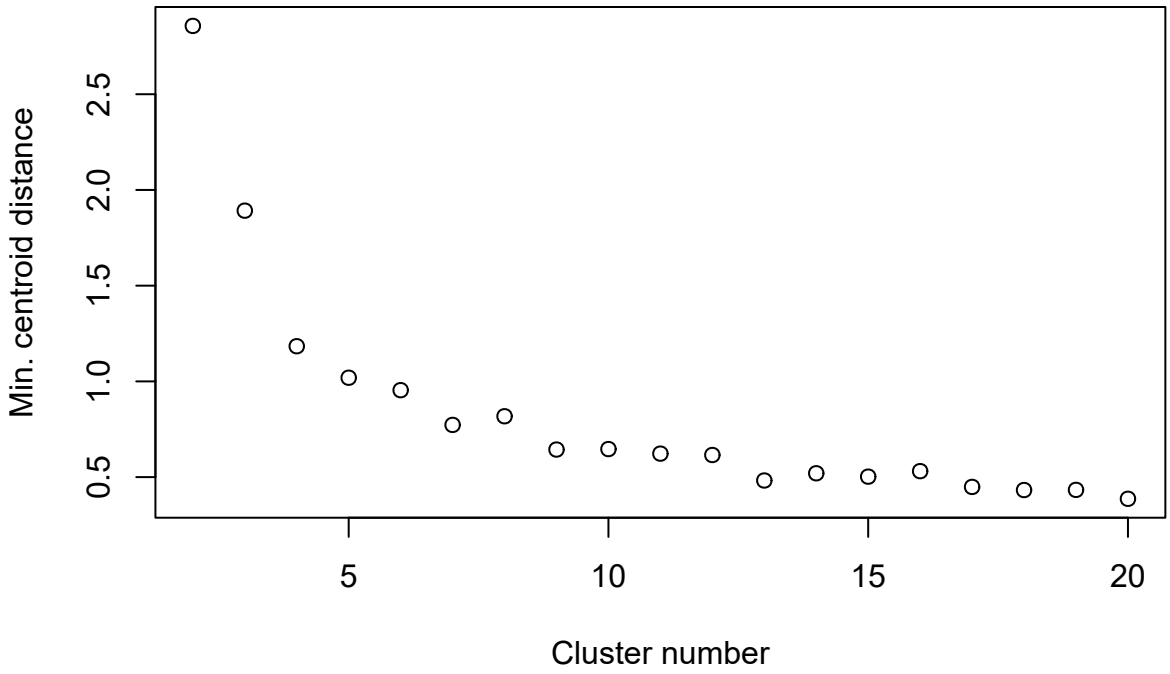
Load data

```
# consensus differentially expressed genes
DEGs <- readRDS('results/bulkRNAseq_mmus_DEGs.rds')

# RNAseq data
RNAseq <- readRDS('results/bulkRNAseq_mmus_data_filt_norm.rds')
```

Clustering of expression profiles

```
DEGs_union <- lapply(DEGs$filt[4:length(DEGs$filt)], function(x) x$ensembl_gene_id) %>%
  unlist() %>%
  unique()

eset <- RNAseq$tpm %>%
  groupTransform(group.lbls = RNAseq$design_meta$condition,
                 FUN = function(x) apply(x,1,mean)) %>%
  dplyr::filter(row.names(.) %in% DEGs_union) %>%
  dplyr::select(CDm, HFDm, DPN, DIP, E2, PPT)

# zscore data (mean=0, sd=1)
eset <- new('ExpressionSet', exprs = as.matrix(eset)) %>%
  Mfuzz::standardise()
```

```
# estimate fuzzifier parameter for clustering
m_eset <- Mfuzz::mestimate(eset)

# determine cluster number with minimum centroid distance
Mfuzz::Dmin(eset, m = m_eset, crange = seq(2,20,1), repeats = 5)
```




```
##  [1] 2.8568062 1.8918404 1.1836708 1.0191843 0.9541434 0.7726632 0.8178600
##  [8] 0.6440237 0.6464875 0.6226779 0.6155064 0.4823358 0.5200341 0.5024704
## [15] 0.5311577 0.4487651 0.4324951 0.4333430 0.3866989
```

```
set.seed(2)

# generate mfuzz clusters (n=4)
clusters <- mfuzz(eset, c = 4, m = m_eset)

# check correlation of cluster centroids
cor(t(clusters[[1]]))
```

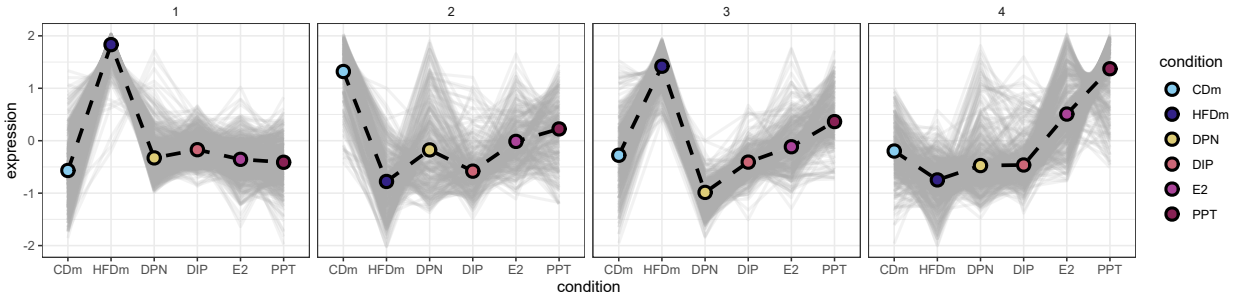
```
##            1          2         3         4
## 1  1.00000000  0.02779203  0.3290034 -0.4890136
## 2  0.02779203  1.00000000 -0.3195364  0.8166390
## 3  0.32900336 -0.31953640  1.0000000 -0.6242888
## 4 -0.48901360  0.81663896 -0.6242888  1.0000000
```

```
# get cluster membership values of genes
cluster_memberships <- acore(eset, cl = clusters, min.acore = 0.0)

# assign to cluster with top membership value
cluster_memberships <- do.call(rbind,
                               lapply(seq_along(cluster_memberships),
                                      function(x) {data.frame(CLUSTER=x,
                                                              cluster_memberships[[x]])})) %>%
  dplyr::mutate(CLUSTER=dplyr::recode(CLUSTER, !!!setNames(c(4,3,2,1), seq(1,4,1))))

# check number of genes per cluster
table(cluster_memberships$CLUSTER)
```

```
##
##   1   2   3   4
```

```
## 577 258 295 347
```

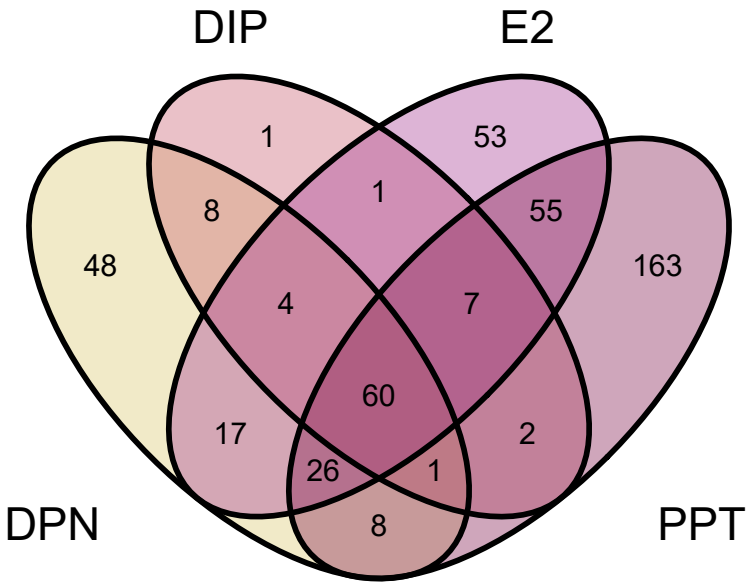
```
DEG_clusters <- list()

# extract gene profiles and cluster assignments
DEG_clusters$genes <- as.data.frame(exprs(eset)) %>%
  tibble::rownames_to_column(var = 'geneID') %>%
  tibble::add_column(GeneSymbol = .$geneID, .after = 'geneID') %>%
  dplyr::mutate(GeneSymbol=dplyr::recode(GeneSymbol,
                                         !!!setNames(RNAseq$annotation$external_gene_name,
                                                     RNAseq$annotation$geneID))) %>%
  merge(cluster_memberships, by.x = 'geneID', by.y = 'NAME', sort = F)

# extract cluster centroid profiles
DEG_clusters$centroids <- as.data.frame(clusters$centers) %>%
  tibble::add_column(geneID = paste0('centroid_', c(4,3,2,1)), .before = 'CDm') %>%
  tibble::add_column(GeneSymbol = paste0('centroid_', c(4,3,2,1)), .before = 'CDm') %>%
  dplyr::mutate(CLUSTER=c(4,3,2,1),
                MEM.SHIP=1) %>%
  arrange(CLUSTER)

df <- DEG_clusters$genes %>%
  dplyr::bind_rows(DEG_clusters$centroids) %>%
  tidyr::pivot_longer(cols = c('CDm', 'HFDm', 'DPN', 'DIP', 'E2', 'PPT'),
                      names_to = 'condition',
                      values_to = 'expression') %>%
  dplyr::mutate(CLUSTER=factor(CLUSTER, levels = 1:4),
                condition=factor(condition, levels = c('CDm', 'HFDm', 'DPN', 'DIP', 'E2', 'PPT')))

ggplot(df, aes(x=condition, y=expression, color=CLUSTER, group=geneID, fill=condition)) +
  geom_line(data = subset(df, !grepl('centroid', GeneSymbol)), size = 1,  color=alpha('#AEAEAE', 0.15)) +
  geom_line(data = subset(df, grepl('centroid', GeneSymbol)), size = 1.2, color='black', linetype='dashed') +
  geom_point(data = subset(df, grepl('centroid', GeneSymbol)), shape=21, size=3, stroke=1.5, color='black') +
  scale_fill_manual(values = colPals$conditions) +
  facet_wrap(~CLUSTER, nrow = 1) +
  theme_bw() +
  theme(strip.background = element_blank())
```



Intersection of ER-agonist treatment DEG sets

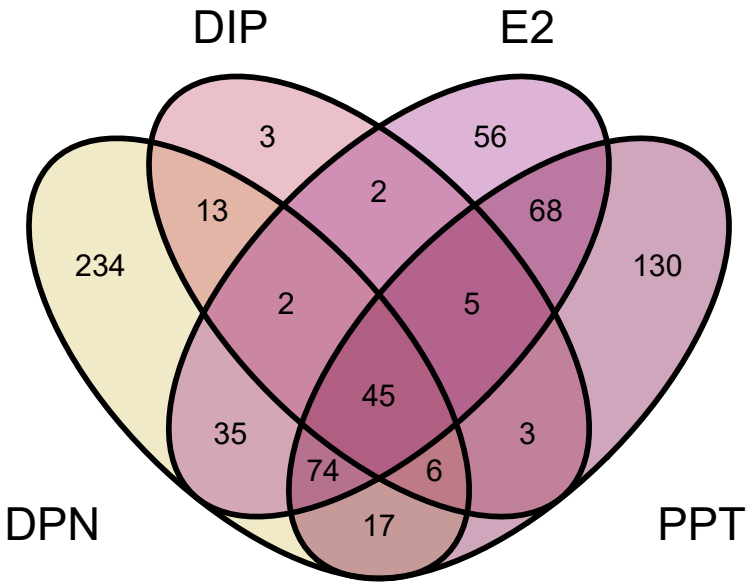
```
# upregulated genes
DEGs_ER_agonist_up <- lapply(DEGs$filt[5:8], function(x){
  x %>%
    dplyr::filter(log2FoldChange>0) %>%
    dplyr::pull(ensembl_gene_id)
})
names(DEGs_ER_agonist_up) <- c('DPN','DIP','E2','PPT')
ggvenn::ggvenn(DEGs_ER_agonist_up,
       columns = c('DPN','DIP','E2','PPT'),
       fill_color = c('#DDCC77', '#CC6677', '#AA4499', '#882255'),
       fill_alpha = 0.4,
       show_percentage = F) +
  ggtitle('Upregulated')
```

Upregulated




```
# downregulated genes
DEGs_ER_agonist_down <- lapply(DEGs$filt[5:8], function(x){
  x %>%
    dplyr::filter(log2FoldChange<0) %>%
    dplyr::pull(ensembl_gene_id)
})
names(DEGs_ER_agonist_down) <- c('DPN','DIP','E2','PPT')
ggvenn::ggvenn(DEGs_ER_agonist_down,
       columns = c('DPN','DIP','E2','PPT'),
       fill_color = c('#DDCC77', '#CC6677', '#AA4499', '#882255'),
       fill_alpha = 0.4,
       show_percentage = F) +
  ggtitle('Downregulated')
```

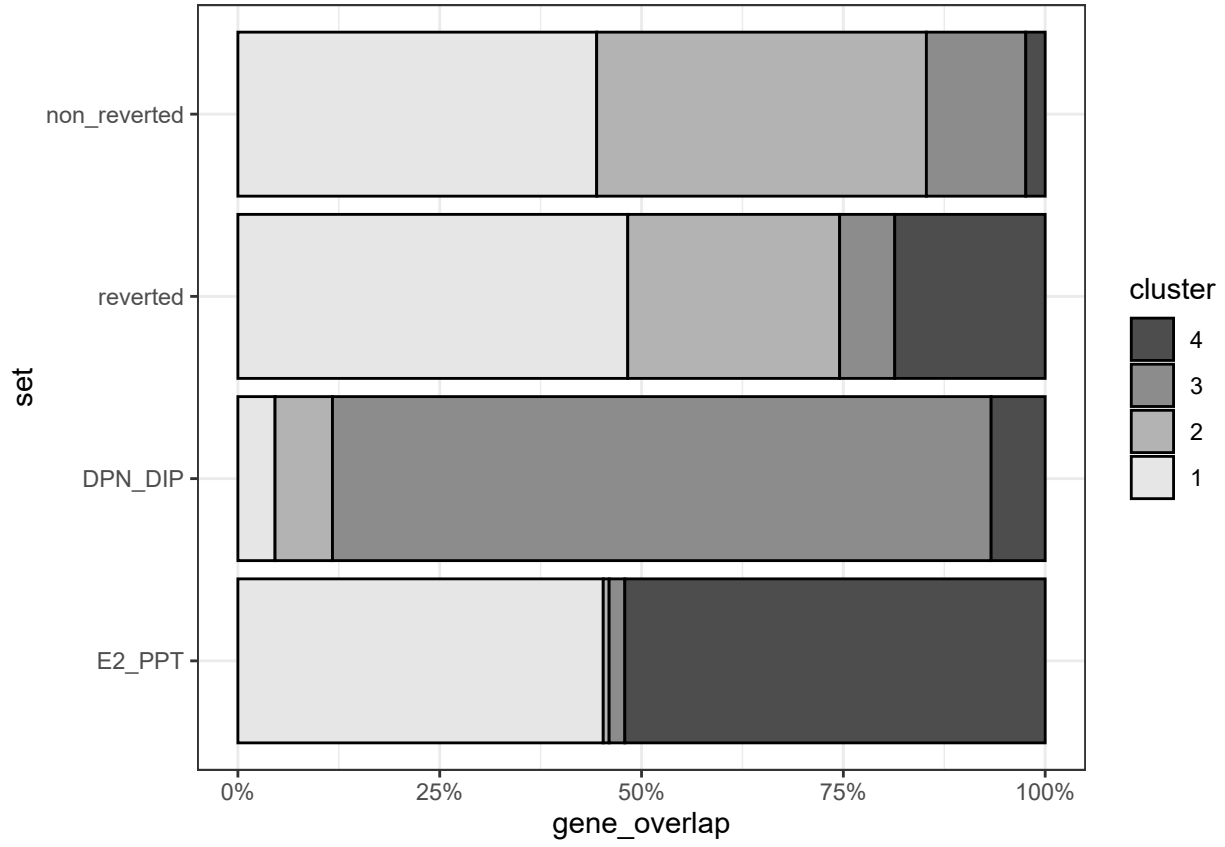
Downregulated



Analysis of relevant DEG sets

```
# extract relevant intersections of DEGs sets between conditions
DEG_sets <- list()

DEG_sets$gene_id$non_reverted <- dplyr::setdiff(DEGs$filt$CDmVsHFDm$ensembl_gene_id,
                                                c(DEGs$filt$DPNVsHFDm$ensembl_gene_id,
                                                  DEGs$filt$DIPVsHFDm$ensembl_gene_id,
                                                  DEGs$filt$E2VsHFDm$ensembl_gene_id,
                                                  DEGs$filt$PPTVsHFDm$ensembl_gene_id))

DEG_sets$gene_id$reverted <- dplyr::intersect(DEGs$filt$CDmVsHFDm$ensembl_gene_id,
                                              c(DEGs$filt$DPNVsHFDm$ensembl_gene_id,
                                                DEGs$filt$DIPVsHFDm$ensembl_gene_id,
                                                DEGs$filt$E2VsHFDm$ensembl_gene_id,
                                                DEGs$filt$PPTVsHFDm$ensembl_gene_id))

DEG_sets$gene_id$DPN_DIP <- dplyr::setdiff(unique(c(DEGs$filt$DPNVsHFDm$ensembl_gene_id,
                                                    DEGs$filt$DIPVsHFDm$ensembl_gene_id)),
                                           c(DEGs$filt$CDmVsHFDm$ensembl_gene_id,
                                             DEGs$filt$E2VsHFDm$ensembl_gene_id,
                                             DEGs$filt$PPTVsHFDm$ensembl_gene_id))

DEG_sets$gene_id$E2_PPT <- dplyr::setdiff(c(DEGs$filt$E2VsHFDm$ensembl_gene_id,
                                            DEGs$filt$PPTVsHFDm$ensembl_gene_id),
                                          c(DEGs$filt$CDmVsHFDm$ensembl_gene_id,
                                            DEGs$filt$DPNVsHFDm$ensembl_gene_id,
                                            DEGs$filt$DIPVsHFDm$ensembl_gene_id)) %>%
  unique()

# get gene symbols
DEG_sets$gene_symbols <- lapply(DEG_sets$gene_id, function(x) {
  dplyr::recode(x,
                !!!setNames(RNAseq$annotation$external_gene_name,
                            RNAseq$annotation$geneID)) %>%
    unique()
```

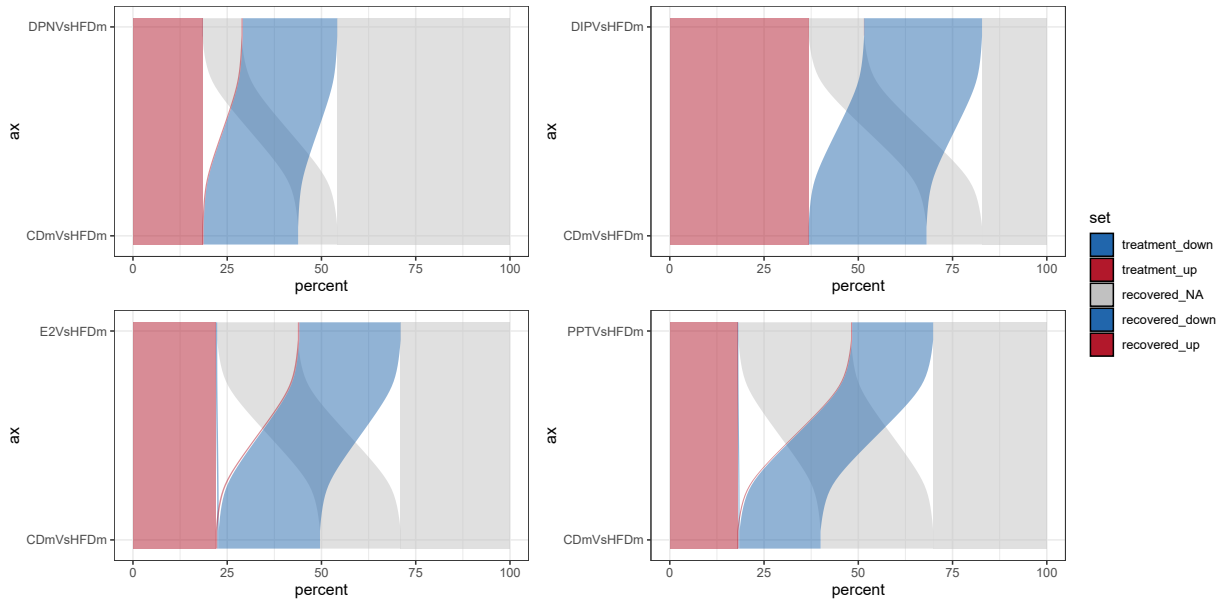
```
})

# count genes per expression cluster for each set
comb <- expand.grid(names(DEG_sets$gene_id), seq(1,4))
comb <- split(comb, 1:nrow(comb))

df <- lapply(comb, function(x) {
  genes.x <- DEG_sets$gene_id[[x[[1]]]]
  genes.y <- DEG_clusters$genes %>%
    dplyr::filter(CLUSTER==x[[2]]) %>%
    dplyr::pull(geneID)

  data.frame(set=x[[1]],
             cluster=x[[2]],
             gene_overlap=dplyr::intersect(genes.x, genes.y) %>% length())
}) %>%
  dplyr::bind_rows() %>%
  mutate(cluster=factor(cluster, levels = rev(seq(1,4))))

ggplot(df, aes(x=set, y=gene_overlap, fill=cluster)) +
  geom_bar(position='fill', stat='identity', color='black', size=0.5) +
  scale_y_continuous(labels=scales::percent_format()) +
  scale_fill_manual(values = colPals$clusters) +
  scale_x_discrete(limits = rev) +
  coord_flip() +
  theme_bw()
```



Recovery of gene expression by different ER agonist treatments

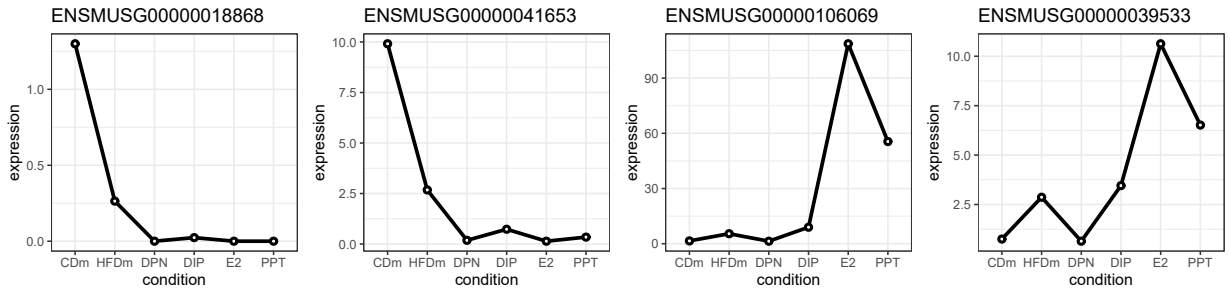
```
ER_reverted <- lapply(list(DPN='DPNVsHFDm',DIP='DIPVsHFDm',E2='E2VsHFDm',PPT='PPTVsHFDm'), function(x) {

  treatment_up <- DEGs$filt[[x]] %>%
    dplyr::filter(log2FoldChange>0) %>%
    dplyr::pull(ensembl_gene_id)
  treatment_down <- DEGs$filt[[x]] %>%
```

```
    dplyr::filter(log2FoldChange<0) %>%
    dplyr::pull(ensembl_gene_id)
  recovered_up <- DEGs$filt$CDmVsHFDm %>%
    dplyr::filter(ensembl_gene_id %in% DEG_sets$gene_id$reverted
                  & log2FoldChange>0
                  & ensembl_gene_id %in% DEGs$filt[[x]]$ensembl_gene_id) %>%
    dplyr::pull(ensembl_gene_id)
  recovered_down <- DEGs$filt$CDmVsHFDm %>%
    dplyr::filter(ensembl_gene_id %in% DEG_sets$gene_id$reverted
                  & log2FoldChange<0
                  & ensembl_gene_id %in% DEGs$filt[[x]]$ensembl_gene_id) %>%
    dplyr::pull(ensembl_gene_id)

  df <- data.frame(matrix(nrow = 0, ncol = 3))
  df <- rbind(df, c('treatment_up', 'recovered_up', length(intersect(treatment_up, recovered_up))))
  df <- rbind(df, c('treatment_up', 'recovered_down', length(intersect(treatment_up, recovered_down))))
  df <- rbind(df, c('treatment_up', 'recovered_NA', length(setdiff(treatment_up, c(recovered_up, recovered_down)))))
  df <- rbind(df, c('treatment_down', 'recovered_up', length(intersect(treatment_down, recovered_up))))
  df <- rbind(df, c('treatment_down', 'recovered_down', length(intersect(treatment_down, recovered_down))))
  df <- rbind(df, c('treatment_down', 'recovered_NA', length(setdiff(treatment_down, c(recovered_up, recovered_down)))))
  colnames(df) <- c(x, 'CDmVsHFDm', 'gene_overlap')
  df$gene_overlap <- as.numeric(df$gene_overlap)
  df$percent <- df$gene_overlap/sum(df$gene_overlap)*100

  df <- df %>%
    ggalluvial::to_lodes_form(key = 'ax', value = 'set', id = 'overlap', axes = 1:2) %>%
    dplyr::mutate(ax=factor(ax, levels = c('CDmVsHFDm',x)),
                  set=factor(set, levels = c('treatment_down','treatment_up','recovered_NA','recovered_down','recovered_up')))

  df
})

p <- lapply(ER_reverted, function(df) {

  ggplot(df, aes(x=ax, y = percent, stratum=set, alluvium=overlap, fill=set)) +
  ggalluvial::geom_alluvium(width = 1/12) +
  ggalluvial::geom_stratum(width = 1.5/12) +
  scale_x_discrete(expand = c(.05, .05)) +
  scale_fill_manual(values = c(treatment_up='#B2182B',
                               treatment_down='#2166AC',
                               recovered_up='#B2182B',
                               recovered_down='#2166AC',
                               recovered_NA='#C1C1C1')) +
  coord_flip() +
  theme_bw()

})

patchwork::wrap_plots(p, nrow=2, ncol=2, byrow=T, guides = 'collect')
```



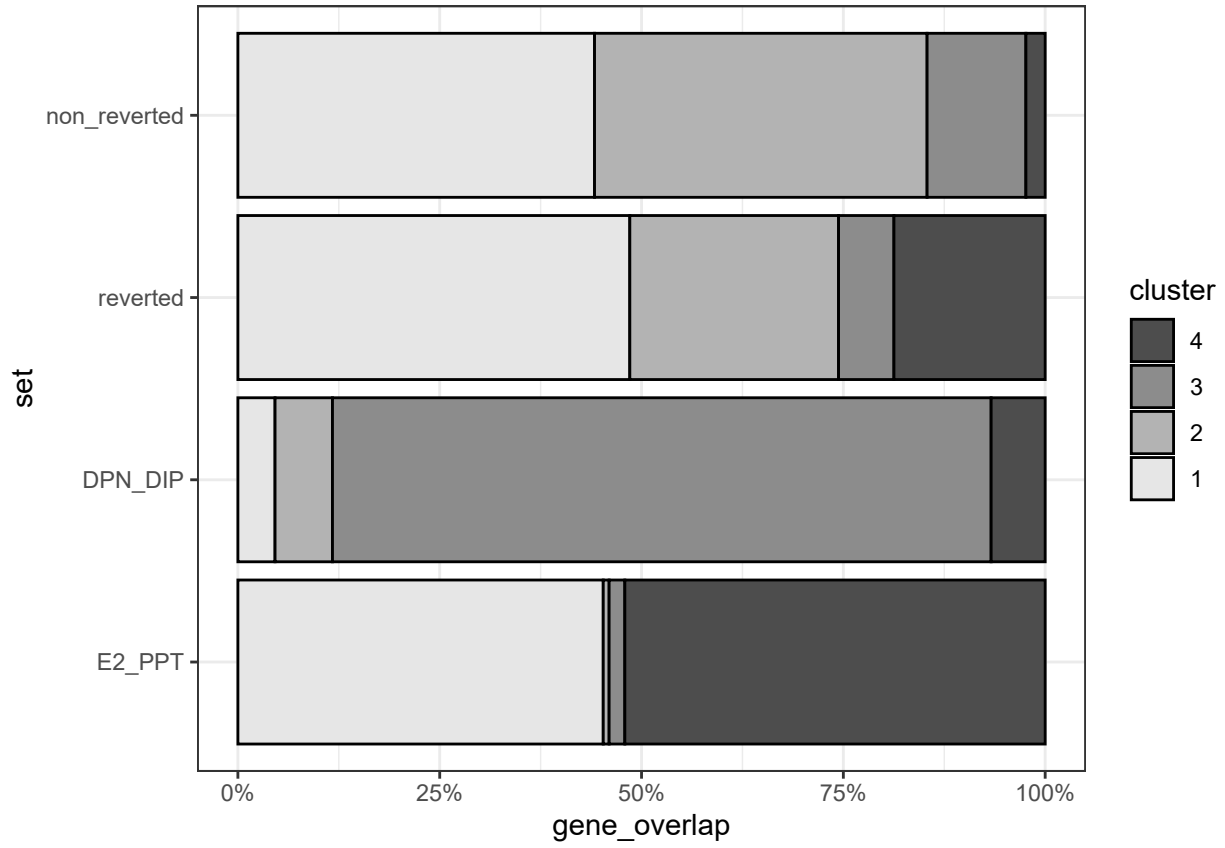
Filter reverted gene set

```
# remove genes that are not truly restored to CD levels by the treatments (see alluvial plots)
recovered_filt <- lapply(list(DPN='DPNVsHFDm',DIP='DIPVsHFDm',E2='E2VsHFDm',PPT='PPTVsHFDm'), function(x) {

  treatment_up <- DEGs$filt[[x]] %>%
    dplyr::filter(log2FoldChange>0) %>%
    dplyr::pull(ensembl_gene_id)
  treatment_down <- DEGs$filt[[x]] %>%
    dplyr::filter(log2FoldChange<0) %>%
    dplyr::pull(ensembl_gene_id)
  recovered_up <- DEGs$filt$CDmVsHFDm %>%
    dplyr::filter(ensembl_gene_id %in% DEG_sets$gene_id$reverted
                  & log2FoldChange>0
                  & ensembl_gene_id %in% DEGs$filt[[x]]$ensembl_gene_id) %>%
    dplyr::pull(ensembl_gene_id)
  recovered_down <- DEGs$filt$CDmVsHFDm %>%
    dplyr::filter(ensembl_gene_id %in% DEG_sets$gene_id$reverted
                  & log2FoldChange<0
                  & ensembl_gene_id %in% DEGs$filt[[x]]$ensembl_gene_id) %>%
    dplyr::pull(ensembl_gene_id)

  c(intersect(treatment_up, recovered_down),
    intersect(treatment_down, recovered_up))

}) %>%
  unlist() %>%
  unique()

recovered_filt <- split(recovered_filt, 1:length(recovered_filt))

p <- lapply(recovered_filt, function(x) {
  df <- RNAseq$tpm %>%
    groupTransform(group.lbls = RNAseq$design_meta$condition,
                   FUN = function(x) apply(x,1,mean)) %>%
    dplyr::filter(row.names(.) %in% x) %>%
    dplyr::select(CDm, HFDm, DPN, DIP, E2, PPT) %>%
    tidyr::pivot_longer(cols = dplyr::everything(), names_to = 'condition', values_to = 'expression') %>%
    dplyr::mutate(condition = factor(condition, levels = c('CDm', 'HFDm', 'DPN', 'DIP', 'E2', 'PPT')))

  ggplot(df, aes(x=condition, y=expression)) +
    geom_line(size = 1.2, group=1) +
    geom_point(shape=21, size=1, stroke=1.5, fill='white') +
    ggtitle(x) +
    theme_bw() +
    theme(strip.background = element_blank())

})

patchwork::wrap_plots(p, nrow=1, ncol=4, byrow=T)
```




```
# update DEG sets
# ENSMUSG00000018868 and ENSMUSG00000041653 do not show recovery with any treatment
DEG_sets$gene_id$non_reverted <- c(DEG_sets$gene_id$non_reverted, c('ENSMUSG00000018868', 'ENSMUSG00000041653'))
DEG_sets$gene_id$reverted <- dplyr::setdiff(DEG_sets$gene_id$reverted, c('ENSMUSG00000018868', 'ENSMUSG00000041653'))

DEG_sets$gene_symbols <- lapply(DEG_sets$gene_id, function(x) {
  dplyr::recode(x,
                !!!setNames(RNAseq$annotation$external_gene_name,
                            RNAseq$annotation$geneID)) %>%
    unique()
})
```

```
# replot updated DEG sets
# count genes per expression cluster for each set
comb <- expand.grid(names(DEG_sets$gene_id), seq(1,4))
comb <- split(comb, 1:nrow(comb))

df <- lapply(comb, function(x) {
  genes.x <- DEG_sets$gene_id[[x[[1]]]]
  genes.y <- DEG_clusters$genes %>%
    dplyr::filter(CLUSTER==x[[2]]) %>%
    dplyr::pull(geneID)

  data.frame(set=x[[1]],
             cluster=x[[2]],
             gene_overlap=dplyr::intersect(genes.x, genes.y) %>% length())
}) %>%
  dplyr::bind_rows() %>%
  mutate(cluster=factor(cluster, levels = rev(seq(1,4))))

ggplot(df, aes(x=set, y=gene_overlap, fill=cluster)) +
  geom_bar(position='fill', stat='identity', color='black', size=0.5) +
  scale_y_continuous(labels=scales::percent_format()) +
  scale_fill_manual(values = colPals$clusters) +
  scale_x_discrete(limits = rev) +
  coord_flip() +
  theme_bw()
```



Gene ontology analysis

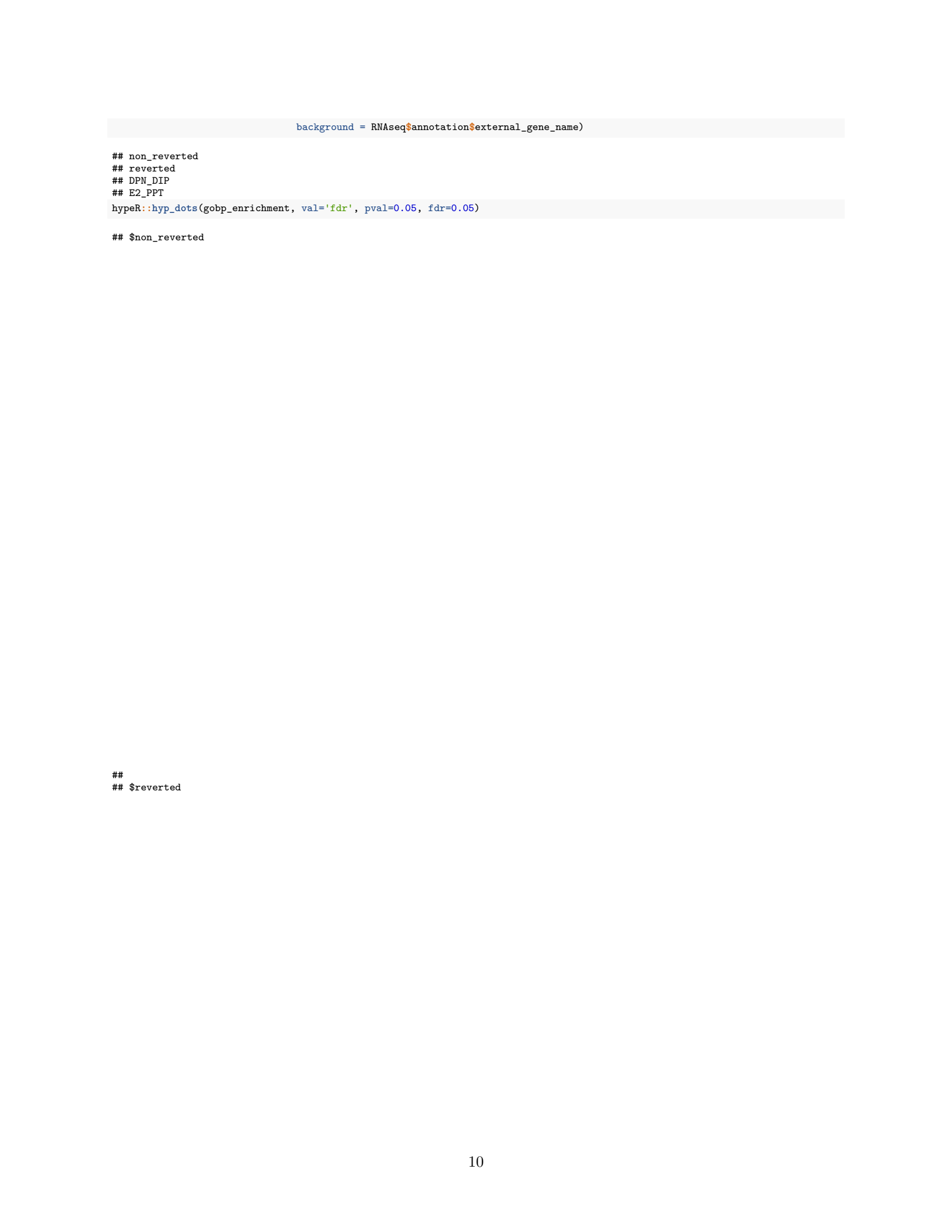
```
# load MGI GO biological process annotation
mgi_gobp <- readGMT('data/mgi_jul2021_gobp_annotation.gmt')

# perform overrepresentation analysis for gene sets changed by HFD and treatments
gobp_enrichment <- hypeR::hypeR(signature = DEG_sets$gene_symbols,
                                genesets = mgi_gobp$genesets,
                                test = 'hypergeometric',
```

```
                                background = RNAseq$annotation$external_gene_name)
```

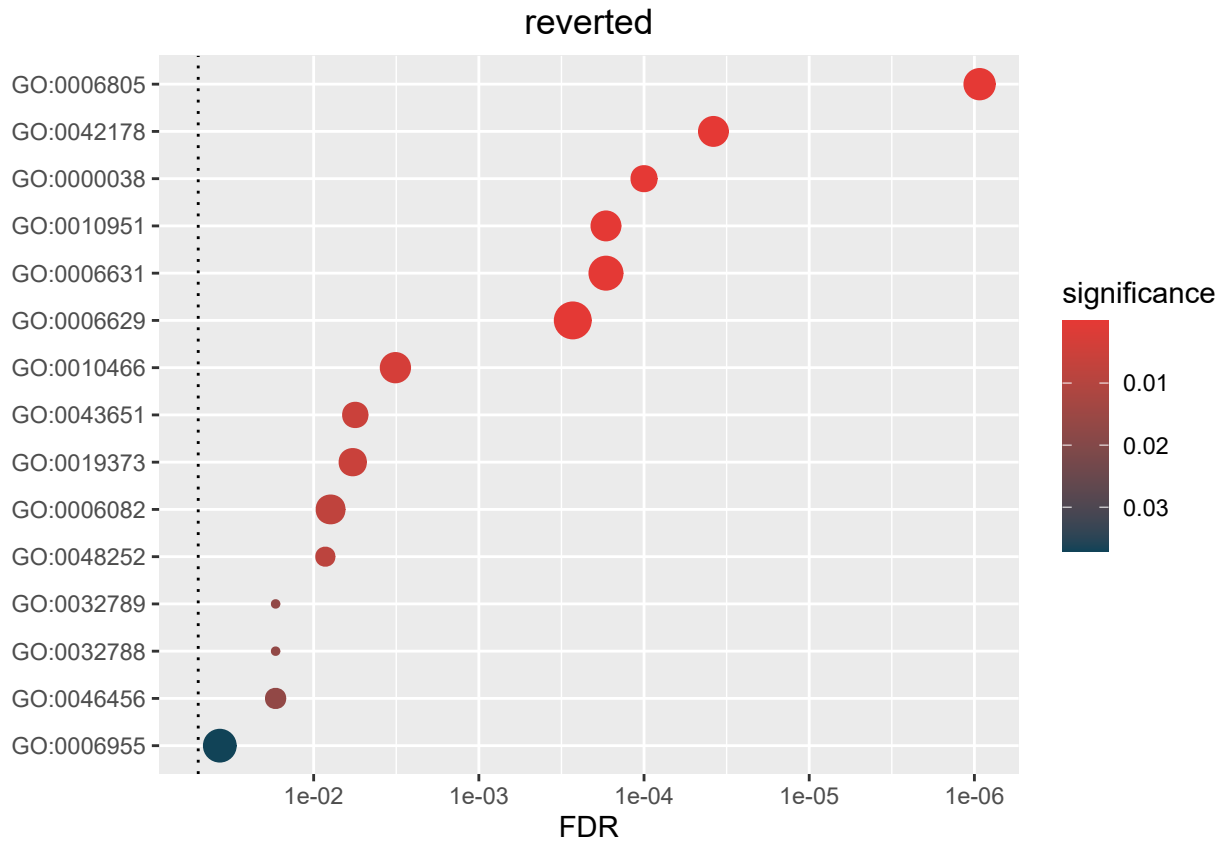
```
## non_reverted
## reverted
## DPN_DIP
## E2_PPT
```

```
hypeR::hyp_dots(gobp_enrichment, val='fdr', pval=0.05, fdr=0.05)
```

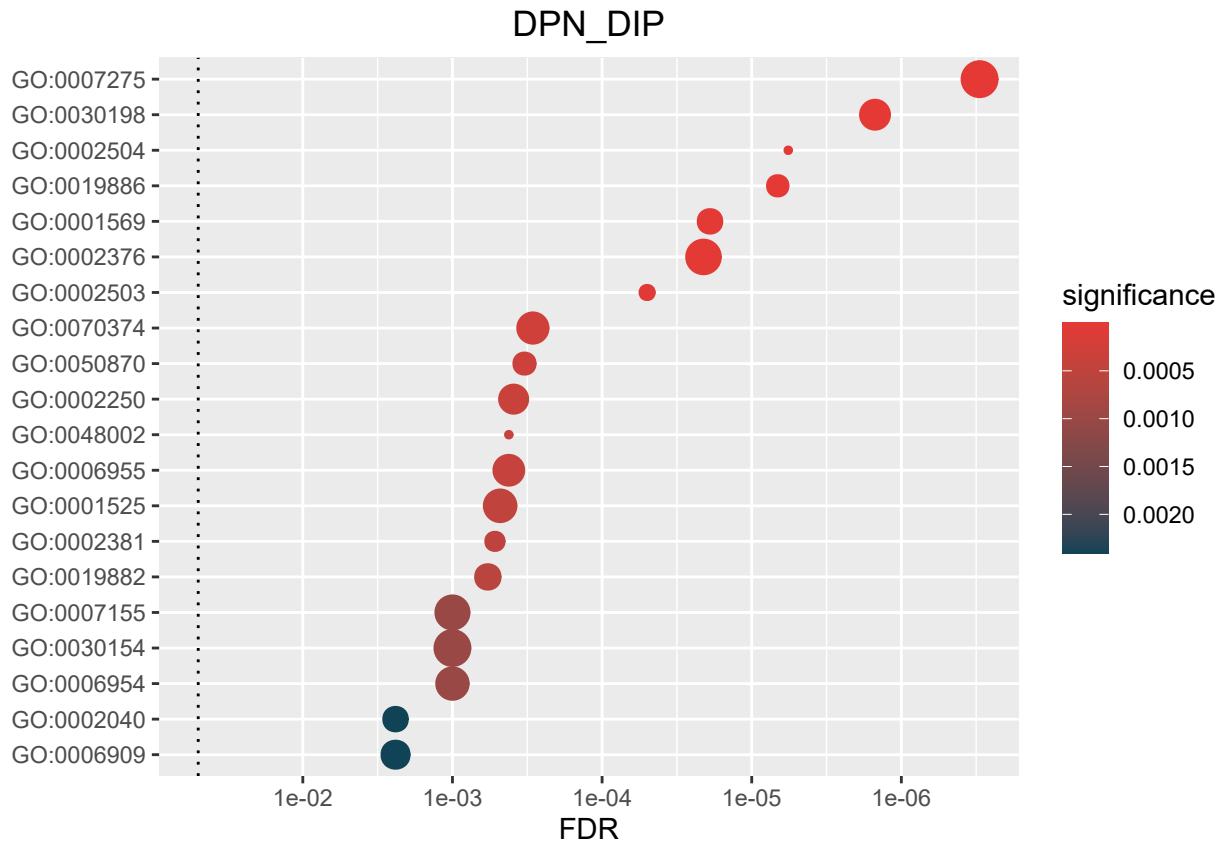
```
## $non_reverted
```



```
##
## $reverted
```


reverted
GO:0006805
GO:0042178
GO:0000038
GO:0010951
GO:0006631
GO:0006629
GO:0010466
GO:0043651
GO:0019373
GO:0006082
GO:0048252
GO:0032789
GO:0032788
GO:0046456
GO:0006955
1e-02
1e-03
1e-04
1e-05
1e-06
FDR
significance
0.01
0.02
0.03

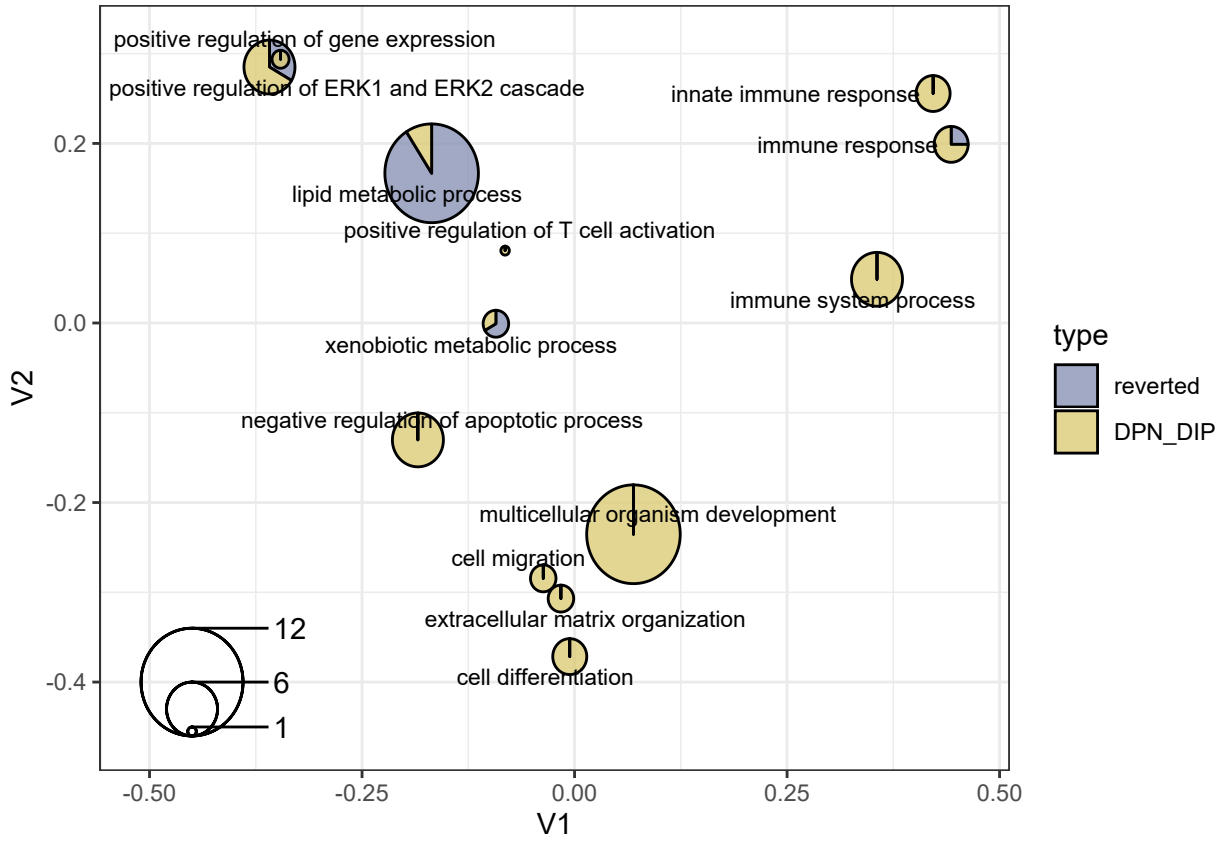
```
##
## $DPN_DIP
```


DPN_DIP
GO:0007275
GO:0030198
GO:0002504
GO:0019886
GO:0001569
GO:0002376
GO:0002503
GO:0070374
GO:0050870
GO:0002250
GO:0048002
GO:0006955
GO:0001525
GO:0002381
GO:0019882
GO:0007155
GO:0030154
GO:0006954
GO:0002040
GO:0006909
1e-02
1e-03
1e-04
1e-05
1e-06
FDR
significance
0.0005
0.0010
0.0015
0.0020

```
##
## $E2_PPT
```

```r
# append descriptions and filter
gobp_enrichment <- lapply(gobp_enrichment$data, function(x) {
  x$data %>%
    dplyr::mutate(description=dplyr::recode(label,
                                            !!!setNames(mgi_gobp$geneset.descriptions,
                                                        mgi_gobp$geneset.names))) %>%
    dplyr::filter(fdr<0.05)
})

# collapse GO terms based on similarity
set.seed(5)

gobp_terms <- lapply(names(gobp_enrichment), function(x) {
  gobp_enrichment[[x]] %>%
    dplyr::mutate(set=x) %>%
    dplyr::rename(goid=label) %>%
    dplyr::select(set, goid)
}) %>%
  dplyr::bind_rows()

sim_mat <- rrvgo::calculateSimMatrix(gobp_terms$goid %>% unique(),
                                     orgdb='org.Mm.eg.db',
                                     ont='BP',
                                     method='Wang')

reduced_terms <- rrvgo::reduceSimMatrix(sim_mat,
                                        scores = NULL,
                                        threshold=0.9,
                                        orgdb='org.Mm.eg.db')

gobp_enrichment_reduced <- rrvgo::scatterPlot(sim_mat, reduced_terms)$data %>%
  tibble::rownames_to_column(var = 'goid') %>%
  dplyr::right_join(gobp_terms, by = 'goid') %>%
  dplyr::mutate(term=make.unique(term))

# get x and y coordinates in semantic space for parent terms
sim_mat <- rrvgo::calculateSimMatrix(gobp_enrichment_reduced$parent %>% unique(),
                                     orgdb='org.Mm.eg.db',
                                     ont='BP',
                                     method='Wang')
```

```
reduced_terms <- rrvgo::reduceSimMatrix(sim_mat,
                                        scores = NULL,
                                        threshold=0,
                                        orgdb='org.Mm.eg.db')

df <- gobp_enrichment_reduced %>%
  dplyr::group_by(set, parentTerm) %>%
  dplyr::summarise(set=set,
                   parentTerm=parentTerm,
                   n=n()) %>%
  unique() %>%
  dplyr::left_join(rrvgo::scatterPlot(sim_mat, reduced_terms)$data %>%
                     dplyr::select(parentTerm, V1, V2),
                   by = 'parentTerm') %>%
  tidyr::pivot_wider(names_from = 'set',
                     values_from = 'n',
                     values_fill = 0) %>%
  dplyr::mutate(size=reverted+DPN_DIP)

ggplot(data=df, aes(x=V1, y=V2)) +
  scatterpie::geom_scatterpie(data=df, aes(x=V1, y=V2, r=size*0.005),
                              cols=c('reverted','DPN_DIP'), color='black') +
  scatterpie::geom_scatterpie_legend(df$size*0.005, x=-0.45, y=-0.4, n=3, labeller=function(x) x*200) +
  ggrepel::geom_text_repel(aes(label = parentTerm), size = 3) +
  scale_fill_manual(values = alpha(c('#8A94B7', '#DDCC77'), 0.8)) +
  theme_bw()
```

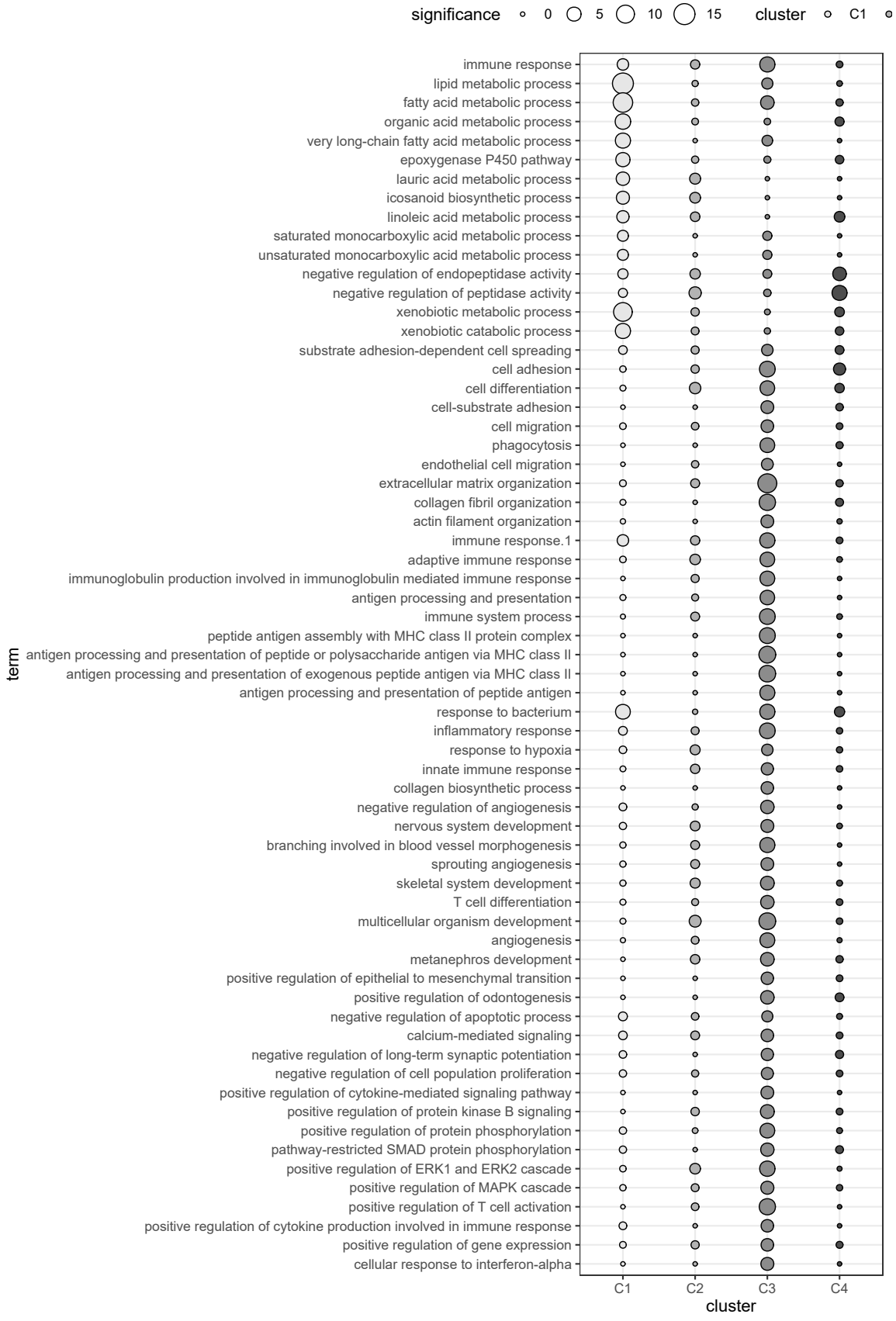



```
# enrichment of GO terms across DEG clusters
cl_sets <- lapply(setNames(seq(1,4), c('C1','C2','C3','C4')), function(x) {
  DEG_clusters$genes %>%
    dplyr::filter(CLUSTER==x) %>%
    dplyr::pull(GeneSymbol)
})

# get enrichments
cl_gobp_enrichment <- hypeR::hypeR(signature = cl_sets,
                                   genesets = mgi_gobp$genesets,
                                   test = 'hypergeometric',
```

```
                                    background = RNAseq$annotation$external_gene_name)
```

```
## C1
## C2
## C3
## C4
```

```
cl_gobp_enrichment <- lapply(cl_gobp_enrichment$data, function(x) {
  x$data %>%
    dplyr::mutate(description=dplyr::recode(label,
                                            !!!setNames(mgi_gobp$geneset.descriptions,
                                                        mgi_gobp$geneset.names)))
})

df <- lapply(names(cl_gobp_enrichment), function(x) {
  cl_gobp_enrichment[[x]] %>%
    dplyr::mutate(cluster=x) %>%
    dplyr::rename(goid=label)
}) %>%
  dplyr::bind_rows() %>%
  dplyr::inner_join(gobp_enrichment_reduced %>%
                      dplyr::select(set, goid, term, parentTerm),
                    by = 'goid') %>%
  dplyr::mutate(significance=-log10(pval)) %>%
  dplyr::arrange(parentTerm) %>%
  dplyr::arrange(factor(set, levels = c('reverted','DPN_DIP'))) %>%
  dplyr::mutate(term=factor(term, levels = unique(term)))

ggplot(df, aes(x=cluster, y=term, fill=cluster, size=significance)) +
  geom_point(shape=21, color='black', stroke=0.5) +
  scale_size_continuous(guide='legend', limits = c(0,15), range = c(1, 6), breaks = c(0,5,10,15)) +
  scale_fill_manual(values = setNames(colPals$clusters, paste0('C',seq(1,4))), guide='legend') +
  scale_y_discrete(limits = rev) +
  theme_bw() +
  guides(fill=guide_legend(order = 2), size=guide_legend(order = 1)) +
  theme(
    legend.position = 'top',
    legend.justification = 'left'
  )
```


significance
0
5
10
15
cluster
C1
immune response
lipid metabolic process
fatty acid metabolic process
organic acid metabolic process
very long-chain fatty acid metabolic process
epoxygenase P450 pathway
lauric acid metabolic process
icosanoid biosynthetic process
linoleic acid metabolic process
saturated monocarboxylic acid metabolic process
unsaturated monocarboxylic acid metabolic process
negative regulation of endopeptidase activity
negative regulation of peptidase activity
xenobiotic metabolic process
xenobiotic catabolic process
substrate adhesion-dependent cell spreading
cell adhesion
cell differentiation
cell-substrate adhesion
cell migration
phagocytosis
endothelial cell migration
extracellular matrix organization
collagen fibril organization
actin filament organization
immune response.1
adaptive immune response
immunoglobulin production involved in immunoglobulin mediated immune response
antigen processing and presentation
immune system process
peptide antigen assembly with MHC class II protein complex
antigen processing and presentation of peptide or polysaccharide antigen via MHC class II
antigen processing and presentation of exogenous peptide antigen via MHC class II
antigen processing and presentation of peptide antigen
response to bacterium
inflammatory response
response to hypoxia
innate immune response
collagen biosynthetic process
negative regulation of angiogenesis
nervous system development
branching involved in blood vessel morphogenesis
sprouting angiogenesis
skeletal system development
T cell differentiation
multicellular organism development
angiogenesis
metanephros development
positive regulation of epithelial to mesenchymal transition
positive regulation of odontogenesis
negative regulation of apoptotic process
calcium-mediated signaling
negative regulation of long-term synaptic potentiation
negative regulation of cell population proliferation
positive regulation of cytokine-mediated signaling pathway
positive regulation of protein kinase B signaling
positive regulation of protein phosphorylation
pathway-restricted SMAD protein phosphorylation
positive regulation of ERK1 and ERK2 cascade
positive regulation of MAPK cascade
positive regulation of T cell activation
positive regulation of cytokine production involved in immune response
positive regulation of gene expression
cellular response to interferon-alpha
term
C1
C2
C3
C4
cluster

Export

```
saveRDS(DEG_sets, file = 'results/bulkRNAseq_mmus_DEG_sets.rds')
saveRDS(gobp_enrichment, file = 'results/bulkRNAseq_mmus_DEG_sets_GOBP_enrichment.rds')
```

SessionInfo

```
sessionInfo()
```

```
## R version 4.0.5 (2021-03-31)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
##  [1] grid      tcltk     parallel  stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] ggrepel_0.9.1       scatterpie_0.1.5    rrvgo_1.2.0
##  [4] hypeR_1.4.0         patchwork_1.1.1     ggalluvial_0.12.3
##  [7] ggvenn_0.1.8        Mfuzz_2.50.0        DynDoc_1.66.0
## [10] widgetTools_1.66.0  e1071_1.7-4         Biobase_2.48.0
## [13] BiocGenerics_0.36.1 forcats_0.5.1       stringr_1.4.0
## [16] dplyr_1.1.2         purrr_0.3.4         readr_2.1.2
## [19] tidyr_1.2.0         tibble_3.2.1        ggplot2_3.3.3
## [22] tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##   [1] colorspace_2.0-0      ellipsis_0.3.2        class_7.3-18
##   [4] fs_1.5.2              rstudioapi_0.13       farver_2.0.3
##   [7] bit64_4.0.5           AnnotationDbi_1.52.0  fansi_0.4.2
##  [10] lubridate_1.8.0       xml2_1.3.3            cachem_1.0.3
##  [13] GOSemSim_2.14.2       knitr_1.31            polyclip_1.10-0
##  [16] jsonlite_1.8.0        gridBase_0.4-7        broom_0.8.0
##  [19] GO.db_3.11.4          dbplyr_2.1.1          pheatmap_1.0.12
##  [22] ggforce_0.3.2         shiny_1.6.0           BiocManager_1.30.21.1
##  [25] msigdbr_7.5.1         compiler_4.0.5        httr_1.4.2
##  [28] rvcheck_0.1.8         backports_1.4.1       assertthat_0.2.1
##  [31] fastmap_1.1.0         cli_3.6.1             org.Mm.eg.db_3.12.0
##  [34] later_1.1.0.1         tweenr_1.0.1          visNetwork_2.0.9
##  [37] htmltools_0.5.2       tools_4.0.5           igraph_1.2.6
##  [40] NLP_0.2-1             gtable_0.3.3          glue_1.4.2
##  [43] Rcpp_1.0.7            slam_0.1-48           cellranger_1.1.0
##  [46] vctrs_0.6.3           babelgene_22.9        svglite_2.1.0
##  [49] xfun_0.31             openxlsx_4.2.3        rvest_1.0.2
##  [52] mime_0.9              lifecycle_1.0.3       MASS_7.3-53
##  [55] scales_1.2.1          treemap_2.4-2         hms_1.0.0
##  [58] promises_1.1.1        RColorBrewer_1.1-3    yaml_2.2.1
##  [61] memoise_2.0.1         stringi_1.5.3         RSQLite_2.2.3
##  [64] highr_0.10            S4Vectors_0.28.1      zip_2.1.1
##  [67] tkWidgets_1.68.0      rlang_1.1.1           pkgconfig_2.0.3
##  [70] systemfonts_1.0.4     evaluate_0.21         labeling_0.4.2
##  [73] htmlwidgets_1.5.3     bit_4.0.4             tidyselect_1.2.0
##  [76] magrittr_2.0.3        R6_2.5.1              IRanges_2.24.1
##  [79] generics_0.1.3        DBI_1.1.3             pillar_1.9.0
##  [82] haven_2.5.0           withr_2.5.0           reactable_0.2.3
##  [85] modelr_0.1.8          crayon_1.5.1          wordcloud_2.6
##  [88] utf8_1.1.4            tzdb_0.3.0            rmarkdown_2.14
##  [91] readxl_1.4.0          data.table_1.13.6     blob_1.2.4
##  [94] reprex_2.0.1          digest_0.6.27         webshot_0.5.2
##  [97] xtable_1.8-4          tm_0.7-8              httpuv_1.5.5
## [100] stats4_4.0.5          munsell_0.5.0         viridisLite_0.4.2
## [103] kableExtra_1.3.4
```